

AD-A124 250

GENERATING PARALLEL CORRELATED TRANSITION FREQUENCIES  
FOR A MARKOV CHAIN. (U) NORTH CAROLINA UNIV AT CHAPEL  
HILL CURRICULUM IN OPERATIONS R. G S FISHMAN DEC 82  
UNC/ORSA/TR-82/8 N00014-76-C-0302 F/G 12/1

1/1

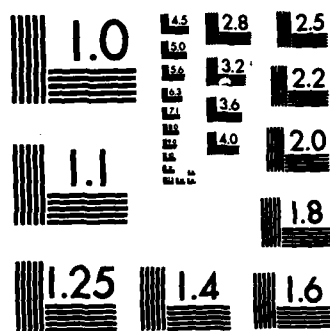
UNCLASSIFIED

NL

END

FILED

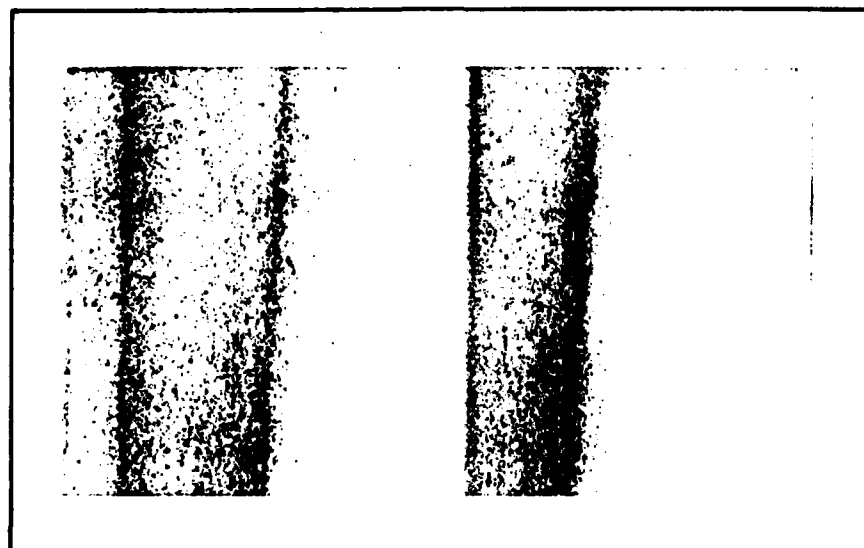
DTIC



MICROCOPY RESOLUTION TEST CHART  
NATIONAL BUREAU OF STANDARDS-1963-A

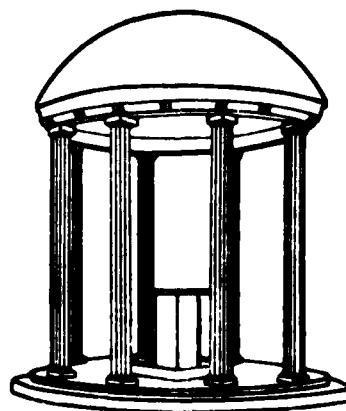
AD A 124250

# OPERATIONS RESEARCH AND SYSTEMS ANALYSIS



UNIVERSITY OF NORTH CAROLINA  
AT CHAPEL HILL

DTIC FILE COPY



DTIC  
ELECTE  
FEB 9 1983  
A

This document has been approved  
for public release and sale; its  
distribution is unlimited.

83 02 09 071

Generating Parallel Correlated Transition

Frequencies for a Markov Chain

George S. Fishman

Technical Report No. UNC/ORSA/TR-82/8

December 1982

DTIC  
ELECTE  
S FEB 9 1983 D  
A

Curriculum in Operations Research

and Systems Analysis

This document has been approved  
for public release and sale; its  
distribution is unlimited.

76  
↓  
This research was supported by the Office of Naval Research under contract  
N00014-26-C-0302. Reproduction in whole or part is permitted for any  
purpose of the United States government.

## Abstract

This paper describes an algorithm and a FORTRAN program called MCHAIN for simulating  $k$  parallel Monte Carlo replications of a Markov chain using rotation sampling. This method of sampling produces  $k$  sample transition frequency vectors with a desirable structure of statistical dependence among them. In particular, these sample vectors can be used to estimate the probability that, say,  $n_{x_1, \dots, x_r}^{s_1 b_1, \dots, s_r b_r}$  transitions of types  $1, \dots, r$  occur during a first passage from state  $a$  to state  $b$  with a variance of the estimate of  $O(1/k^2)$  and a computation time  $O(k)$  as  $k \rightarrow \infty$ . This compares favorably with the case of independent replications wherein the estimate would have a variance  $O(1/k)$  and computation time  $O(k)$  as  $k \rightarrow \infty$ . An example including a sample driver program are presented to illustrate how MCHAIN works in practice.

### Key Words

Markov chain, Monte Carlo methods, rotation sampling, simulation, variance reduction



Accession For	
NTIS GRA&I	<input checked="checked" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
Distribution/	
Availability Codes	
Avail and/or	Special

## Introduction

Let  $p_n = ||p_{ij}||$  denote a positive recurrent aperiodic Markov Chain with state space  $S = \{0, 1, \dots, n\}$ . Let  $N_{ij}$  denote the number of one-step transitions that occur from state  $i$  to state  $j$  during a first passage from state  $a$  to state  $b$   $a, b \in S$  on an arbitrarily selected replication and let  $N_{ij}^{(\ell)}$  denote this quantity observed on replication  $\ell$ . The purpose of this paper is to describe an algorithm and a FORTRAN program called MCHAIN that generates  $k$  sample data sets  $\{N_{ij}^{(1)}; i, j \in S\}, \dots, \{N_{ij}^{(k)}; i, j \in S\}$  from the chain  $p_n$  with a specialized dependence among sets that allows one to estimate quantities of interest with greater statistical efficiency than independent data sets allow.

As an example of the use of these data sets, suppose one wants to compute

$$p(n_1, \dots, n_r) = \text{probability that during a first passage} \quad (1)$$

$$\text{from } a \text{ to } b \quad n_1, \dots, n_r \text{ one-step transitions}$$

$$\text{occur from } i_1 \text{ to } j_1, \dots, i_r \text{ to } j_r, \text{ respectively.}$$

For Markov chains with relatively general structure, convenient analytical representations are not available for (1) so that computation is not possible. In fact no convenient representations are available for chains with special structure but arbitrary  $r > 1$ . One way to overcome this absence of representation is to employ Monte Carlo methods. Assume there are  $k$  simulated data sets  $\{N_{ij}^{(1)}; i, j \in S\}, \dots, \{N_{ij}^{(k)}; i, j \in S\}$  available to estimate (1), each simulation beginning with a departure from state  $a$  and ending upon entry into state  $b$ . Let

$$K_{n_1, \dots, n_r} = \sum_{\ell=1}^k \prod_{m=1}^r \delta(N_{i_m j_m}^{(\ell)} - n_m) \quad (2)$$

where

$$\begin{aligned}\delta(x) &= 1 && \text{if } x = 0 \\ &= 0 && \text{otherwise.}\end{aligned}$$

If  $\{N_{ij}^{(\ell)}; i, j \in S\}$   $\ell = 1, \dots, k$  are independent, then

$$\hat{p}(n_1, \dots, n_r) = \frac{N_{n_1, \dots, n_r}^{(k)}}{k} \quad (3)$$

is an unbiased estimator of  $p(n_1, \dots, n_r)$  with variance proportional to  $1/k$ . Using a specialized sampling procedure called rotation sampling, MCHAIN induces statistical dependence among the data sets  $\{N_{ij}^{(\ell)}; i, j \in S\}$   $\ell = 1, \dots, k$  that preserves the unbiasedness of (3) but changes the convergence rate of its variance to  $O(1/k^2)$ . Fishman (1982b) illustrates the significance of this improved convergence for the estimation of the probability density function of first passage time in a semi-Markov process. More generally, the improved convergence is clearly beneficial when estimating functions of Markov chains that are linear combinations of  $p(n_1, \dots, n_r)$  for varying  $n_1, \dots, n_r$ . Complete details about rotation sampling can be found in Fishman (1982a).

Section 1 describes how one constructs a simulation model to induce the desired dependence and shows its consequences in terms of the improved efficiency. For comparison a procedure is first described for simulating independent replications of a Markov chain. Section 2 describes MCHAIN and, in particular, its input requirements. Section 3 presents an example of how MCHAIN performs.

## 1. Serial and Parallel Simulation

To put the proposed sampling scheme into perspective we first describe the more conventional simulation of a Markov chain in which data on  $k$  independent replications are collected in serial order.

### Serial Simulation

1. Start in state  $i=a$  .
2. For  $m=1, \dots, k$  and  $i, j=0, 1, \dots, n$ :  $N_{ij}^{(m)} \leftarrow 0$  .
3.  $m \leftarrow 1$  .
4. Sample  $U$  from  $u(0,1)$  .  
Use  $U$ ,  $\{p_{ij}; j=0, 1, \dots, n\}$  and the alias sampling method (Walker 1977 and Kronmal and Peterson 1979) to determine entered state  $j$  .
5.  $N_{ij}^{(m)} \leftarrow N_{ij}^{(m)} + 1$  .
6. If  $j \neq b$  ,  $i \leftarrow j$  and go to 4 .
7. If  $m=k$ , deliver  $\{N_{ij}^{(m)}; i, j=0, 1, \dots, n\}$   $m=1, \dots, k$  .
8.  $m \leftarrow m+1$
9.  $i \leftarrow a$  .
10. Go to 4 .

Here  $u(0,1)$  in step 4 denotes the uniform distribution on the half-open interval  $[0,1)$  . The alias sampling method is a procedure for sampling from a tabled discrete distribution at a cost independent of the length of the table. For example, if the entered state is  $i \neq b$  , then sampling occurs from the distribution  $p_{i0}, p_{i1}, \dots, p_{in}$  with constant execution time regardless of the value of  $n$  .

In the case of parallel simulation based on rotation sampling all  $k$  replications begin at the same moment in time. The steps are:

### Parallel Simulation

Definitions:  $i_m$  = state occupied by replication  $m$

$K_j$  = number of replications in state  $j$  at the beginning of a step.

$K_j^*$  = number of replications in state  $j$  at the end of a step.

1. For  $m=1, \dots, k$ :
  - 1a.  $i_m \leftarrow a$ .
  - 1b. For  $i=0, 1, \dots, n$ ;  $j=0, 1, \dots, n$ ,  $N_{ij}^{(m)} \leftarrow 0$ .
2. For  $i=0, 1, \dots, n$ :  $K_i \leftarrow 0$ .
3. For  $i=0, 1, \dots, n$ :  $K_i^* \leftarrow 0$ .
4. Sample  $U_0, U_1, \dots, U_n$  independently from  $U(0,1)$ .
5. For  $m=1, \dots, k$ :
  - 5a. If  $i_m = b$ , go to 5h.
  - 5b. Determine  $j$  using  $U_{i_m}$  and  $\{p_{i_m j}; j=0, 1, \dots, n\}$  with the Fishman-Moore (1982) cutpoint sampling method.
  - 5c.  $K_j^* \leftarrow K_j^* + 1$ .
  - 5d.  $N_{i_m j}^{(m)} \leftarrow N_{i_m j}^{(m)} + 1$ .
  - 5e.  $U_{i_m} \leftarrow U_{i_m} + 1/K_{i_m}$ .
  - 5f. If  $U_{i_m} \geq 1$ ,  $U_{i_m} \leftarrow U_{i_m} - 1$ .
  - 5g.  $i_m \leftarrow j$ .
  - 5h. Continue.
6.  $K_b \leftarrow K_b + K_b^*$ .
7. If  $K_b = k$ , deliver  $\{N_{ij}^{(m)}; i, j=0, 1, \dots, n\}$ .
8. For  $j=1, \dots, k$  and  $j \neq b$ :  $K_j \leftarrow K_j^*$ .
9. Go to 3.

Here rotation sampling is implemented in step 5e so that for  $K_{i_m}$  replications in state  $i_m$  the  $K_{i_m}$  uniform deviates used to determine the newly entered states have the uniform distribution on  $[0,1)$  but are not independent. Fishman (1981, 1982b) contain more detailed descriptions of rotation sampling. Step 5b uses an alternative method to the alias to determine the entered state. In particular, this cutpoint method uses the inverse transform method to select the entered state for replication  $m$  as

$$j = (r: \sum_{s=0}^{r-1} p_{i_m s} \leq U_{i_m} < \sum_{s=0}^r p_{i_m s}, \quad p_{i_m, -1} = 0; \quad r=0,1,\dots,n) \quad (4)$$

at a cost independent of  $n$ . This procedure for sampling from the Markov chain together with the rotation sampling induces the desired dependence. Since the alias sampling method, which is slightly less costly, does not use the inverse transform method, it cannot be used to induce the desired correlation.

To appreciate the significance of the dependence that rotation sampling and the cutpoint method together induce, we first focus on (2). Let  $w=n_1+\dots+n_r$ . Then

$$K_w = \sum_{n_1=0}^w \sum_{n_2=0}^{w-n_1} \dots \sum_{n_r=0}^{w-n_1-\dots-n_{r-1}} K_{n_1, \dots, n_r} \quad (5)$$

is the number of replications absorbed on the  $w$ th transition. Making use of the fact that  $\text{var } K_w = O(1)$ , Proposition 1 in Fishman (1982b) shows that  $\text{var } K_{n_1, \dots, n_r} = O(1)$ . Therefore,  $\text{var } \hat{p}(n_1, \dots, n_r) = O(1/k^2)$ .

### Computation Time Complexity

Although the accelerated convergence of the variances of estimators is important, one also needs to consider how the computation time complexity of the parallel simulation (PS) compares with that of the serial simulation (SS). Both are  $O(K)$ . Although for a given  $k$  parallel simulation takes more time than serial simulation does, the ratio of these times converges to a constant as  $k \rightarrow \infty$ , thereby demonstrating the superiority of parallel simulation as measured by the common variance reduction measure (see Hammersley and Handscomb 1964)

$$VR = \frac{\text{variance using SS}}{\text{variance using PS}} \cdot \frac{\text{computation time using SS}}{\text{computation time using PS}} = O(k) \text{ as } k \rightarrow \infty. \quad (6)$$

### 2. The MCHAIN Program

Figure 1 lists a FORTRAN subroutine called MCHAIN that generates the transition frequency data for a finite-state Markov chain using rotation sampling. The program uses pointers ( $M(*)$ ) to indicate the  $S(*)$

---

Insert Fig. 1 about here.

---

states that can be entered from  $*$ , thus eliminating the nonzero entries in the transition matrix and reducing storage space.

Of special interest is the method of determining the state to be entered by a replication at each transition, as in (4). MCHAIN uses the cutpoint method developed in Fishman and Moore (1981) whose execution time is independent of  $S(J)$ , the number of states that can be entered from  $J$ . Therefore, using this method makes the cost of state transition determination independent of the number of nonzero entries in each row of  $p_n$ . MCHAIN uses the random number generator GGUBS in the IMSL(1982) library, but this can be changed at the user's discretion.

MCHAIN allows for three different levels of output through the variable DETAIL. Setting DETAIL=2 results in a printout of transition frequencies for each replication and summations of these frequencies across replications. Setting DETAIL=1 leads to a printout only of the summations across replications and DETAIL=0 suppresses all printed output. In practice, we envision that MCHAIN may be used to generate input for other simulation, as exemplified in Fishman (1982b). In this case a user needs to alter MCHAIN to store or pass the frequency data and may have limited or no interest in the printout of the frequency data themselves. DETAIL's options enable him to specify his level of interest.

MCHAIN allows for two types of simulation runs, macroreplications and microreplications. MCHAIN runs  $I$  macroreplications each consisting of  $K$  microreplications based on parallel simulation using rotation sampling. The need for these macroreplications arises when estimating the variances of estimators. See Fishman (1982b). In general, it is advisable to make  $K$  large relative to  $I$  to gain the benefit of rotation sampling. Space requirements for the arrays in MCHAIN are roughly  $4 * (SIZE + 8 * NP + 2 * K + 6 * ALL)$  .

### 3. Example

Consider a single server queueing model for which the time between successive arrivals are independent and exponentially distributed with rate  $\lambda$  , service times are independent and exponentially distributed with rate  $\omega$  , the queue discipline is first-come-first-served and there is a finite capacity  $n$  . Corresponding to this formulation is an embedded nearest neighbor Markov chain whose states are the number of

jobs in the system and whose nonzero transition probabilities are

$$\begin{aligned} p_{01} &= 1 \\ p_{i,i-1} &= \frac{\omega}{\lambda + \omega} & i=2, \dots, n \\ p_{i,i+1} &= \frac{\lambda}{\lambda + \omega} & i=1, \dots, n-1 \\ p_{n,n} &= \frac{\lambda}{\lambda + \omega} \end{aligned} \quad (7)$$

We show a run of MCHAIN for a first passage from state  $a=4$  to state  $b=17$  for a single macroreplication containing 15 microreplications based on rotation sampling with  $\lambda=.9$ ,  $\omega=1$  and  $n=19$ . The input for MCHAIN is  $NP=20$ ,  $INITIAL=4$ ,  $ABSORB=17$ ,  $I=1$ ,  $K=15$ ,  $SEED=1556203872$ ,  $SIZE=1000$  and  $DETAIL=2$ . The quantities  $ALL$ ,  $KK$  and  $N$  and the  $M$ ,  $P$ ,  $S$  and  $SUMS$  arrays are computed in the sample driver program in Fig. 2. Figure 3

---

Insert Figs. 2 and 3 about here.

---

shows the output for MCHAIN. The entries in the table are the frequencies of transitions for all possible transition type by microreplication.

## References

- Fishman, G.S. (1981) "Accelerated Accuracy in the Simulation of Markov Chains," Technical Report No. UNC/ORSA/TR-81/1, Curriculum in Operations Research and Systems Analysis, University of North Carolina at Chapel Hill. To appear in Operations Research.
- Fishman, G.S. and L.R. Moore, III (1981). "Sampling from a Discrete Distribution While Preserving Monotonicity," Technical Report No. UNC/ORSA/TR-81/7, Curriculum in Operations Research and Systems Analysis, University of North Carolina at Chapel Hill.
- Fishman, G.S. (1982a). "Simulating a Markov Chain with a Superefficient Sampling Method," Technical Report No. UNC/ORSA/TR-82/3, Curriculum in Operations Research and Systems Analysis, University of North Carolina at Chapel Hill. To appear in Operations Research.
- Fishman, G.S. (1982b). "Superefficient Simulation of Markov Chains and Semi-Markov Processes," Technical Report No. UNC/ORSA/TR-82/5 Curriculum in Operations Research and Systems Analysis, University of North Carolina at Chapel Hill.
- Hammersley, J.M. and D.C. Handscomb (1964). Monte Carlo Methods, Methuen.
- International Mathematical and Statistical Libraries, Inc. (1982). IMSL Library, Houston, Texas.
- Kronmal, Richard A. and Arthur V. Peterson (1979). "On the Alias Method for Generating Random Variables from a Discrete Distribution," The American Statistician, 4, 214-218.
- Walker, A.J. (1977). "An Efficient Method of Generating Discrete Random Variables with General Distributions," ACM Transactions on Mathematical Software, 3, 253-256.

```

C ***
C *** SUBROUTINE MCHAIN
C ***
C ***
C *** THIS SUBROUTINE SIMULATES K REPLICATIONS OF AN (N+1)-STATE
C *** MARKOV CHAIN IN PARALLEL USING ROTATION SAMPLING.
C ***
C *** FOR REFERENCE SEE:
C ***
C *** FISHMAN, G. S. (1982). "SUPEREFFICIENT SIMULATION OF MARKOV
C *** CHAINS AND SEMI-MARKOV PROCESSES," REPORT UNC/ORSA/TR 82/5
C *** AND "GENERATING PARALLEL CORRELATED TRANSITION FREQUENCIES
C *** FOR A MARKOV CHAIN," REPORT UNC/ORSA/TR 82/8, CURRICULUM
C *** IN OPERATIONS RESEARCH AND SYSTEMS ANALYSIS , UNIVERSITY
C *** OF NORTH CAROLINA AT CHAPEL HILL.
C ***
      SUBROUTINE MCHAIN (NP,INITAL,ABSORB,P,Q,M,S,SUMS,ALL,KK,I,K,
1      SEED,SIZE,UU,V,DETAIL,CUTP,ROTATE,CN,KSTAR,KPRIME,STATE)
C ***
C *** DESCRIPTION OF VARIABLES:
C ***
C *** ABSORB      = ABSORBING STATE
C *** ALL         = SUM OF S(J) FOR ALL J
C *** CN(*)       = HOLDS TRANSITION COUNTS FOR EACH TRANSITION TYPE AND
C ***              EACH MICROREPLICATION; LAST COLUMN CCNTAINS TOTALS
C ***              BY TRANSITION TYPE FOR ALL MICROREPLICATIONS
C *** COLMAX      = MAXIMUM NUMBER OF COLUMNS TO PRINT PER OUTPUT PAGE
C *** CCUNT       = UNIFORM DEVIATE COUNTER
C *** CUTP(*)     = POINTERS FOR CUTPOINT SAMPLING METHOD
C ***              SEE FISHMAN, G. S. AND L. R. MOORE, III (1981).
C ***              "SAMPLING FROM A DISCRETE DISTRIBUTION WHILE
C ***              PRESERVING MONOTONICITY", TECHNICAL REPORT 81/7,
C ***              OPERATIONS RESEARCH AND SYSTEMS ANALYSIS,
C ***              UNIVERSITY OF NORTH CAROLINA AT CHAPEL HILL.
C *** DETAIL      = DETAIL FLAG, VALUES:
C ***              0 FOR NO PRINTING, 1 FOR SUMMARY TOTALS ONLY,
C ***              2 FOR FULL DETAIL PRINTING
C *** LSEED       = RANDOM NUMBER GENERATOR SEED (DOUBLE PRECISION)
C *** EP          = STEP VARIABLE FOR SAMPLING TRANSITION TYPE
C *** GGUBS       = IMSL SUBROUTINE FOR RANDOM NUMBER GENERATION
C *** I           = NUMBER OF MACROREPLICATIONS
C *** I1          = INDEX USED BY OUTPUT SECTION
C *** IFIRST      = NUMBER OF FIRST COLUMN TO PRINT ON CURRENT OUTPUT PAGE
C *** II          = INDEX USED BY OUTPUT SECTION
C *** IIS         = S (J+1)
C *** ILAST       = NUMBER OF LAST COLUMN TO PRINT ON CURRENT OUTPUT PAGE
C *** IND         = FLAG, INITIALLY SET TO 0 TO ALLOW FIRST TRANSITION
C ***              IF STARTING IN ABSORBING STATE
C *** INITAL      = INITIAL STATE
C *** IP          = INDEX USED BY OUTPUT SECTION

```

Fig. 1 MCHAIN FORTRAN Program

```

C *** ISEED      = SAVED INITIAL SEED
C *** IX         = INDEX
C *** J          = INDEX FOR STATES
C *** J1         = INDEX FOR MICROREPLICATIONS
C *** JA         = INDEX FOR ALL
C *** JJ         = INDEX USED FOR INDIRECT SUBSCRIPTING
C *** K          = NUMBER OF PARALLEL MICROREPLICATIONS
C *** K1         = K+1
C *** KA         = NUMBER OF MICROREPLICATIONS THAT HAVE
C ***           = RETURNED TO ABSORBING STATE
C *** KPRIME(*)  = NUMBER IN STATE AT END OF TRANSITION
C *** KSTAR(*)   = NEXT KPRIME(*)
C *** KTEST      = TEMPORARY VARIABLE USED IN OUTPUT SECTION
C *** KTOTAL     = TOTAL NUMBER OF TRANSITIONS
C *** LCOUNT    = INDEX USED BY OUTPUT SECTION TO COUNT OUTPUT LINES
C *** LI         = NUMBER OF CURRENT MACROREPLICATION
C *** LMAX       = MAXIMUM NUMBER OF LINES TO PRINT PER OUTPUT PAGE
C *** LX         = CUTOPOINT TO ACCELERATE SAMPLING PROCEDURE
C *** M(*)       = STATES THAT CAN BE REACHED FROM J-1
C *** N          = NUMBER OF HIGHEST STATE (NP-1)
C *** NP         = NUMBER OF STATES
C *** NSKIP      = NUMBER OF OUTPUT PAGES TO BE PRINTED
C *** CM         = USED IN SETTING UP STATE TRANSITION PROBABILITIES
C *** P(*)       = TRANSITION MATRIX
C *** Q(*)       = VECTOR OF CUMULATIVE PROBABILITIES FOR EACH TRANSITION TYPE
C *** R          = INDEX FOR STATES
C *** ROTATE(*)  = ROTATION INDEX FOR SAMPLING FROM EACH STATE
C *** S(J)       = NUMBER OF STATES THAT CAN BE ENTERED FROM J-1
C *** SEED       = RANDOM NUMBER GENERATOR SEED
C *** SIZE       = BLOCK SIZE FOR RANDOM NUMBER GENERATION
C *** SJ         = S(J)
C *** SK         = SUMS(J)
C *** STATE(*)   = CURRENT STATE OCCUPIED BY MICROREPLICATION J1
C *** SUMS(J)    = SUM OF S(1) THRU S(J-1), AND SUMS(1)=0
C *** U          = CURRENT UNIFORM DEVIATE
C *** UU(*)      = CURRENT UNIFORM DEVIATE FOR TRANSITION TYPE *
C *** V(*)       = UNIFORM DEVIATE ARRAY
C ***
C      INTEGER CUTP(ALL), ABSORB, ALL, CN(KK), COUNT, DETAIL, KPRIME(NP), I,
1      I1, IND, INITAL, ISEED, IX, J, J1, JA, JJ, K, KK, K1, K, KKA, LI, LX, LCOU
2      LMAX, N(ALL), KSTAR(NP), NP, R, S(NP), SEED, SIZE, SJ, SK,
3      STATE(K), SUMS(NP)
C      INTEGER CCLMAX, IPFIRST, I1, IIS, ILAST, IP, ITOTAL, KTEST, KTOTAL, N, NSKIP
C      REAL      V(SIZE)
C      DOUBLE PRECISION  ECTATE(NP), EP, OM, UU(NP), U, DSEED, P(ALL), Q(ALL)
C ***
C ***      INITIALIZE DOUBLE PRECISION SEED
C ***
C      DSEED=SEED
C      K1=K+1

```

Fig. 1 (Continued)

```

C ***
C *** DETERMINE TRANSITION PROBABILITIES
C ***
      DO 100 J=1,NP
        SK = SUMS(J)
        Q(SK+1) = P(SK+1)
        IF(S(J).LT.2) GO TO 100
        SJ = S(J)
        DO 90 R=2,SJ
          Q(SK+R) = Q(SK+R-1) + P(SK+R)
        90   CONTINUE
        Q(SK+SJ) = 1.0D0
      100  CONTINUE
C ***
C *** SET UP INDIRECT INDEXING FOR TRANSITION ARRAYS
C ***
      DO 200 J=1,NP
        SJ=S(J)
        EP=(1.0/SJ)
        OM=0
        LX=SUMS(J)+1
        DO 200 R=1,SJ
          150   CUTP(SUMS(J)+R)=LX
                IF(Q(LX).GT.OM) GO TO 195
                LX=LX+1
                GO TO 150
          195   OM=OM+EP
        200   CONTINUE
      DO 950 LI=1,I
C ***
C *** SET STARTING SEED
C ***
        ISEED=DSEED
C ***
C *** SET FLAG TO ALLOW INITIAL TRANSITION FROM ABSORBING STATE
C ***
        IND=0
C ***
C *** GET ARRAY OF RANDOM DEVIATES
C ***
        CALL GGUES(ISEED,SIZE,V)
C ***
C *** GET INITIAL RANDOM DEVIATE
C ***
        UU(INITAL+1)=V(1)
        U=V(1)
        COUNT=2
C ***
C *** START MICROREPLICATIONS IN INITIAL STATE
C *** AND INITIALIZE TRANSITION COUNTERS

```

Fig. 1 (Continued)

```

C ***
      KPRIME (INITAL+1)=K
      DO 210 J1=1,K
        STATE (J1)=INITAL
        DO 210 JA=1,ALL
          CN (ALL*(J1-1)+JA)=0
210    CONTINUE
      DO 215 J=1,NP
        ROTATE (J)=0.0D0
215    KSTAR (J)=0
      KA=0
C ***
C *** BEGIN SIMULATION OF STATE TRANSITIONS
C ***
      DO 220 J1=1,K
C ***
C *** DC NOT MOVE FROM ABSORBING STATE EXCEPT DURING INITIAL TRANSITION
C ***
        IF (STATE (J1).EQ.ABSORB.AND.IND.EQ.1) GO TO 300
        IX=STATE (J1)+1
C ***
C *** TEST TO PREVENT DIVISION BY ZERO
C ***
        IF (KPRIME (IX).EQ.0) GO TO 300
C ***
C *** GET RANDOM DEVIATE FOR THIS MICROREPLICATION
C ***
        U=UU (IX)+(ROTATE (IX)/KPRIME (IX))
        IF (U.GE.1.0) U=U-1.
C ***
C *** FIND CUTPOINT FOR SEARCH OF SAMPLING DISTRIBUTION
C ***
        LX=(S (IX)*U)
C ***
C *** DETERMINE THE DIRECTION OF TRANSITION
C ***
        JJ=CUTP (SUMS (IX)+LX+1)
225    IF (U.LE.Q (JJ)) GO TO 250
        JJ=JJ+1
        GO TO 225
C ***
C *** INCREMENT THE APPROPRIATE TRANSITION COUNTER
C ***
250    CN (ALL*(J1-1)+JJ)=CN (ALL*(J1-1)+JJ)+1
C ***
C *** SET THE STATE FOR MICROREPLICATION J1
C ***
        STATE (J1)=H (JJ)
C ***
C *** INCREMENT THE TEMPORARY COUNTER AND

```

Fig. 1 (Continued)

```

C *** THE ACTIVE MICROREPLICATION COUNTER
C ***
      KSTAR(M(JJ)+1) = KSTAR(M(JJ)+1)+1
      ROTATE(IX) = ROTATE(IX)+1.0
300    CONTINUE
C ***
C *** SET FLAG TO PREVENT TRANSITIONS FROM ABSORBING STATE
C ***
      IND=1
C ***
C *** ADD NEWLY ABSORBED MICROREPLICATIONS TO COUNTER
C ***
      KA=KA+KSTAR(ABSORB+1)
C ***
C *** IF ALL K MICROREPLICATIONS HAVE BEEN ABSORBED, EXIT
C ***
      IF(KA.EQ.K) GC TO 500
      KSTAR(ABSORB+1) = KA
C ***
C *** REINITIALIZE COUNTERS FOR NEXT SET OF TRANSITIONS
C ***
      DO 400 J=1,NP
        ROTATE(J) = 0.0D0
        KPRIME(J) = KSTAR(J)
        KSTAR(J) = 0
        IF(KPRIME(J) * (J-ABSORB-1).EQ.0) GO TO 400
C ***
C *** GET NEXT RANDOM DEVIATE
C ***
      UU(J) = V(COUNT)
      CCOUNT = COUNT+1
      IF(COUNT.LE.SIZE) GO TO 400
C ***
C *** IF NECESSARY, GET NEW ARRAY OF RANDOM DEVIATES
C ***
      CALL GGUBS(DSEED,SIZE,V)
      COUNT=1
400    CONTINUE
      GO TO 220
500    SEED=DSEED
      IF(DETAIL.EQ.0) GC TO 990
C ***
C *** PRINT RESULTS OF SIMULATION
C ***
      WRITE(3,9000) LI
9000  FORMAT('1'/' RESULTS FROM MCHAIN FOR MACROREPLICATION NO.',I5//)
550  WRITE(3,9010) NP,INITAL,ABSORB,ALL,K,I,ISEED,SEED,SIZE
9010  FORMAT(//,
1      ' NO. OF STATES                      =',I10//,
2      ' INITIAL STATE                      =',I10//,

```

Fig. 1 (Continued)

```

3          ' ABSORBING STATE                      =',I10//,
4          ' TOTAL NO. OF (I,J) PAIRS              =',I10//,
5          ' NO. OF CORRELATED MICROREPLICATIONS   =',I10//,
6          ' NO. OF INDEPENDENT MACROREPLICATIONS  =',I10//,
7          ' INITIAL SEED                          =',I10//,
8          ' FINAL SEED                            =',I10//,
9          ' BLOCKING FACTOR                       =',I10//,
C ***
C *** CALCULATE TOTALS ACROSS ALL CHAINS AND STORE IN CN(*,K1)
C ***
      KTOTAL=0
      DC 650 JA=1,ALL
        ITOTAL = 0
        DO 600 J1=1,K
          ITOTAL = ITOTAL + CN(ALL*(J1-1)+JA)
600      CONTINUE
          CN(ALL*(K1-1)+JA) = ITOTAL
          KTOTAL = KTOTAL + CN(ALL*(K1-1)+JA)
650      CONTINUE
C ***
C *** DETERMINE OUTPUT FORMAT PARAMETERS (PAGE WIDTH AND LENGTH)
C ***
      COLMAX=16
      LMAX=55
      ILAST=0
      IF(DETAIL.NE.1) GO TO 680
C ***
C *** PRINT SUMMARY TOTALS ONLY
C ***
      WRITE(3,2000)
2000     FORMAT('1TRANSITION',6X,'ROW'//,
1         ' FROM TC',6X,'TOTAL'//,
2         ' -----')
      IP=0
      N=NP-1
      LCOUNT=0
      DO 670 J=0,N
        IIS=S(J+1)
        DO 660 II=1,IIS
          LCOUNT=LCOUNT+1
          IF=IP+1
          WRITE(3,2200) J,N(IP),CN(ALL*(K1-1)+IP)
660      CONTINUE
          IF(LCOUNT.LE.LMAX) GO TO 670
          WRITE(3,2000)
          LCCUNT=0
670      CCNTINUE
          GO TO 950
C ***
C *** DETERMINE NUMBER OF OUTPUT PAGES

```

```

C ***
680 IF(K1.GT.COLMAX) GO TO 700
    COLMAX=K1
    NSKIP=1
    GO TO 750
700 NSKIP=K1/CCLMAX
    KTEST=NSKIP*COLMAX
    IF((K1-KTEST).GT.0) NSKIP=NSKIP+1
C ***
C *** PRINT TRANSITION COUNTS IN TABLE FORMAT
C ***
750 DO 900 I1=1,NSKIP
    IFIRST=ILAST+1
    ILAST=I1*COLMAX
    IF(I1.EQ.NSKIP) ILAST=K1
C ***
C *** PRINT PAGE HEADING
C ***
    WRITE(3,2100)
2100 FORMAT(//'1TRANSITION ',42X,'REPLICATES')
    WRITE(3,2110) (IX,IX=IFIRST,ILAST)
2110 FORMAT(' FROM TC ',16(1X,I5))
    WRITE(3,2120)
2120 FORMAT(' -----',
1 '-----'/' )
    IP=0
    N=NF-1
    LCOUNT=0
C ***
C *** PRINT TRANSITION COUNTS
C ***
    DO 850 J=0,N
    IIS=S(J+1)
    DO 800 II=1,IIS
        LCOUNT=LCOUNT+1
        IP=IP+1
        IA=ALL*(IFIRST-1)+IP
        IB=ALL*(ILAST-1)+IP
        WRITE(3,2200) J,M(IP),(CN(IX),IX=IA,IB,ALL)
2200 FORMAT(' ',12,4X,12,' - ',16(1X,I5))
800 CONTINUE
    IF(LCOUNT.LE.LMAX) GO TO 850
    WRITE(3,2100)
    WRITE(3,2110) (IX,IX=IFIRST,ILAST)
    WRITE(3,2120)
    LCCUNT=0
850 CONTINUE
900 CONTINUE
    WRITE(3,2300) K1
2300 FORMAT('0 ***COLUMN ',I5,' CONTAINS THE RCW TOTALS')

```

Fig. 1 (Continued)

-17-

```
950 WRITE(3,2400) KTOTAL
2400 FORMAT('0 ***TOTAL FOR ALL ROWS IS: ',I6)
990 RETURN
END
```

Fig. 1 (Continued)

```

C ***
C *** THIS IS A SAMPLE DRIVER PROGRAM FOR SUBROUTINE MCHAIN
C ***
C *** THIS DRIVER PROGRAM PROVIDES THE SETUP FOR USE OF MCHAIN WITH
C *** THE M/M/1/N QUEUEING MODEL WITH ARRIVAL RATE LAM, SERVICE
C *** RATE W, ONE SERVER AND CAPACITY N. WITH THE EXCEPTION OF
C *** M, P, S AND SUMS, ALL OTHER ARRAYS ARE USED IN MCHAIN.
C ***
C *** DESCRIPTION OF VARIABLES -
C ***
C *** ABSORB      = ABSORBING STATE
C *** ALL         = SUM OF S(J) FOR ALL J
C *** CN(*)       = HOLDS TRANSITION COUNTS FOR EACH TRANSITION TYPE AND
C ***              EACH MICROREPLICATION; LAST COLUMN CONTAINS TOTALS
C *** CUTP(*)     = POINTERS FOR CUTPOINT SAMPLING METHOD
C *** DETAIL      = DETAIL FLAG, VALUES:
C ***              0 FOR NO PRINTING, 1 FOR SUMMARY TOTALS ONLY,
C ***              2 FOR FULL DETAIL PRINTING
C *** I          = DESIRED NUMBER OF INDEPENDENT MACROREPLICATIONS
C *** INITIAL     = INITIAL STATE
C *** J          = INDEX FOR STATES
C *** K          = NUMBER OF PARALLEL MICROREPLICATIONS/MACROREPLICATION
C *** KPRIME(*)   = NUMBER IN STATE AT END OF TRANSITION
C *** KSTAR(*)    = NEXT KPRIME(*)
C *** LAM        = ARRIVAL RATE (<W)
C *** M(*)        = STATES THAT CAN BE REACHED FROM J-1
C *** N          = NUMBER OF HIGHEST STATE (<50)
C *** NP         = NUMBER OF STATES (N+1)
C *** P(*)       = TRANSITION MATRIX
C *** Q(*)       = VECTOR OF CUMULATIVE PROBABILITIES FOR EACH TRANSITION TY
C *** ROTATE(*)   = ROTATION INDEX FOR SAMPLING FROM EACH STATE
C *** SEED       = RANDOM NUMBER GENERATOR SEED
C *** SIZE       = BLOCK SIZE FOR RANDOM NUMBER GENERATOR (<1001)
C *** S(J)       = NUMBER OF STATES THAT CAN BE ENTERED FROM J-1 (<NP)
C *** SUMS(J)    = SUM OF S(1) THRU S(J-1), AND SUMS(1)=0 (<100)
C *** UU(*)      = CURRENT UNIFORM DEVIATE FOR TRANSITION TYPE *
C *** V(*)       = UNIFORM DEVIATE ARRAY
C *** W          = SERVICE RATE
C ***
C      INTEGER CUTP(100),ABSORB,ALL,CN(2500),DETAIL,INITAL,I,J,1,JA,
1      K,KK,KPRIME(50),KSTAR(50),M(100),N,NP,
2      S(50),SEED,SIZE,STATE(50),SUMS(50)
C      REAL    V(1000)
C      DOUBLE PRECISION  RCTATE(50),LAM,P(100),Q(100),UU(50),W
C ***
C *** INITIALIZE VALUES
C ***
C      READ (1,1000) NP,INITAL,ABSORB,I,K,SEED,SIZE
1000  FORMAT (I5/I5/I5/I5/I5/I12/I5)
C      WRITE(3,7060) INITAL,ABSORB

```

Fig. 2 Sample Driver Program

```

7080 FORMAT(' INITIAL STATE:',I5,'      AESORBING STATE:',I5)
      WRITE(3,7081) K,NP
7081 FORMAT(' NUMBER OF REPLICATES:',I5,'      NUMBER OF STATES:',I5)
      WRITE(3,7082) SEED,SIZE
7082 FORMAT(' INITIAL SEED:',I12,'      BLOCKING FACTOR:',I6)
      READ(1,1002) DETAIL
1002 FORMAT(I5)
      READ(1,1001) LAM,W
1001 FORMAT(2F5.2)
      WRITE(3,7085) LAM,W,DETAIL
7085 FORMAT(' LAM: ',F5.2,'      W: ',F5.2,'      DETAIL: ',I3)
C ***
C *** FOR EACH STATE, DETERMINE THE NUMBER OF STATES THAT CAN BE ENTERED
C ***
      S(1)=1
      DO 80 J=2,NP
        S(J)=2
80    CONTINUE
      WRITE(3,2015) (S(J),J=1,NP)
2015  FORMAT(' S(J)  =',20I5)
      SUMS(1)=0
      ALL=S(1)
      DO 90 J=2,NP
        SUMS(J)=ALL
        ALL=ALL+S(J)
90    CONTINUE
      KK=ALL*(K+1)
C *** RESTRICTIONS ON PARAMETER SIZES BELOW ARE FOR THIS PROGRAM.
C ***
C *** COMPUTE TRANSITION PROBABILITIES
C ***
      DO 100 J=2,NP
        P(SUMS(J)+1) = W/(LAM+W)
        P(SUMS(J)+2) = LAM/(LAM+W)
100   CONTINUE
      P(1)=1.0D0
      WRITE(3,2016) (P(JA),JA=1,ALL)
2016  FORMAT(' P(*)  =',20F5.2)
C ***
C *** FOR EACH STATE J DETERMINE STATES THAT CAN BE ENTERED
C ***
      N=NP-1
      DO 400 J=2,N
        M(SUMS(J)+1) = J-2
        M(SUMS(J)+2) = J
400   CONTINUE
      M(1)=1
      M(SUMS(NP)+1) = N-1
      M(SUMS(NP)+2) = N
      WRITE(3,2017) (M(JA),JA=1,ALL)

```

Fig. 2 (Continued)

```
2017 FORMAT(' M(*)  =',20I5)
C ***
C *** CALL SUBROUTINE MCHAIN
C ***
      CALL MCHAIN(NP,INITAL,ABSORB,P,Q,M,S,SUMS,ALL,KK,I,K,
1      SEED,SIZE,UU,V,DETAIL,CUTP,ROTATE,CN,KSTAR,KPRIME,STATE)
C ***
C *** END OF PROGRAM
C ***
      STOP
      END
```

Fig. 2 (Continued)

INITIAL STATE: 4      ABSORBING STATE: 17  
 NUMBER OF REPLICATES: 15      NUMBER OF STATES: 20  
 INITIAL SEED: 1556203872      BLOCKING FACTOR: 1000  
 LAM: 0.95      N: 1.00      DETAIL: 2  
 S(J) = 1      2      2      2      2      2      2      2      2      2      2      2      2      2      2      2      2      2      2  
 P(0) = 1.00      0.51      0.49      0.51      0.49      0.51      0.49      0.51      0.49      0.51      0.49      0.51      0.49      0.51      0.49      0.51      0.49      0.51  
 P(0) = 0.49      0.51      0.49      0.51      0.49      0.51      0.49      0.51      0.49      0.51      0.49      0.51      0.49      0.51      0.49      0.51      0.49      0.51  
 R(0) = 1      0      2      1      3      2      4      3      5      4      6      5      7      6      8      7      9      8      10      9  
 R(0) = 11      10      12      11      13      12      14      13      15      14      16      15      17      16      18      17      19      18      19

RESULTS FROM MCHAIN FOR MACROREPLICATION NO. 1

NO. OF STATES = 20  
 INITIAL STATE = 4  
 ABSORBING STATE = 17  
 TOTAL NO. OF (I,J) PAIRS = 39  
 NO. OF CORRELATED MACROREPLICATIONS = 15  
 NO. OF INDEPENDENT MACROREPLICATIONS = 1  
 INITIAL SEED = 1556203872  
 FINAL SEED = 1950653926  
 BLOCKING FACTOR = 1000

Fig. 3 MCHAIN and DRIVER Program Output

TRANSITION FROM TO		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	1	41	15	0	4	5	20	0	0	24	35	39	26	10	6	0	225
1	0	41	15	0	4	5	20	0	0	24	35	39	26	10	6	0	225
2	1	39	17	2	5	2	16	0	0	24	42	32	28	5	6	0	218
3	2	39	17	2	5	2	16	0	0	24	42	32	28	5	6	0	218
4	3	29	6	3	5	1	11	0	1	20	43	33	32	9	9	1	203
5	4	29	6	3	5	1	11	0	1	20	43	33	32	9	9	1	203
6	5	29	5	1	10	3	4	0	3	18	33	24	39	5	8	1	183
7	6	29	5	1	10	3	4	0	3	18	33	24	39	5	8	1	183
8	7	22	7	5	12	2	8	1	3	17	28	32	23	8	10	0	175
9	8	21	6	4	11	2	7	0	2	16	27	31	22	3	9	2	160
10	9	10	13	3	9	1	3	0	4	10	26	35	16	8	12	2	157
11	10	10	13	3	9	1	3	0	4	10	26	35	16	8	12	2	157
12	11	9	12	2	12	2	11	1	5	9	25	34	15	7	11	3	142
13	12	8	11	2	13	3	10	2	6	8	28	25	6	5	17	3	127
14	13	7	10	3	12	3	11	1	5	7	27	24	5	4	16	3	127
15	14	7	10	3	12	3	11	1	5	7	27	24	5	4	16	3	127
16	15	10	12	1	11	2	13	0	10	3	19	16	10	3	11	5	112
17	16	9	11	0	10	1	12	1	9	2	18	14	9	0	10	6	109
18	17	13	7	2	8	4	10	2	17	1	16	8	7	1	11	6	113
19	18	12	6	1	7	3	9	1	16	0	15	7	6	0	10	5	98
20	19	13	6	1	7	3	9	1	16	0	15	7	6	0	10	5	98
21	20	12	5	2	9	2	10	3	15	0	14	3	3	0	8	4	83
22	21	15	3	1	8	1	5	2	8	2	16	3	2	1	5	6	75
23	22	14	2	0	5	0	4	1	7	1	15	2	1	0	4	5	60
24	23	11	1	3	3	2	7	2	5	4	13	4	2	2	4	7	71
25	24	10	0	2	2	1	6	1	4	3	12	3	1	1	4	6	56
26	25	1	2	6	6	1	7	2	2	2	11	2	1	1	3	2	50
27	26	0	1	5	5	0	6	1	1	1	10	1	0	0	3	1	35
28	27	0	1	4	4	1	5	2	0	1	9	1	0	0	2	2	25
29	28	0	0	3	3	0	4	3	0	1	8	2	0	1	2	1	30
30	29	0	0	2	2	0	3	2	0	0	6	1	0	0	1	0	15
31	30	0	0	1	1	0	2	1	0	0	5	1	0	0	1	0	0
32	31	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
33	32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
34	33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
35	34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
36	35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
37	36	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
38	37	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
39	38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
40	39	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
41	40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

\*\*\*COLUMN 16 CONTAINS THE ROW TOTALS

\*\*\*TOTAL FOR ALL ROWS IS: 3897

Fig. 3 (Continued)

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER UNC/ORSA/TR-82/8	2. GOVT ACCESSION NO. AD-A124250	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) GENERATING PARALLEL CORRELATED TRANSITION FREQUENCIES FOR A MARKOV CHAIN		5. TYPE OF REPORT & PERIOD COVERED TECHNICAL REPORT
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) GEORGE S. FISHMAN		8. CONTRACT OR GRANT NUMBER(s) 76 N00014-26-C-0302
9. PERFORMING ORGANIZATION NAME AND ADDRESS CURRIC. IN OPERATIONS RESEARCH & SYSTEMS ANALYSIS SMITH BUILDING 128 A UNIV. OF NORTH CAROLINA CHAPEL HILL, NC 27514		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS NAVAL ANALYSIS PROGRAM OFFICE OF NAVAL RESEARCH ARLINGTON, VA 22217		12. REPORT DATE DECEMBER 1982
		13. NUMBER OF PAGES 22
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report)  DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)  UNCLASSIFIED		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number)  Markov chain, Monte Carlo methods, rotation sampling, simulation, variance reduction		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number)  This paper describes an algorithm and a FORTRAN program called MCHAIN for simulating <u>k parallel</u> Monte Carlo replications of a Markov chain using <u>rotation sampling</u> . This method of sampling produces <u>k</u> sample transition frequency vectors with a desirable structure of statistical dependence among		

DD FORM 1 JAN 73 1473

EDITION OF 1 NOV 65 IS OBSOLETE  
S/N 0102-LF-014-6601

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

them. In particular, these sample vectors can be used to estimate the probability that, say,  $n_1, \dots, n_r$  transitions of types  $1, \dots, r$  occur during a first passage from state  $a$  to state  $b$  with a variance of the estimate of  $O(1/k^2)$  and a computation time  $O(k)$  as  $k \rightarrow \infty$ . This compares favorably with the case of independent replications wherein the estimate would have a variance  $O(1/k)$  and computation time  $O(k)$  as  $k \rightarrow \infty$ . An example including a sample driver program are presented to illustrate how MCHAIN works in practice.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

END